In the Claims:

1. (Currently amended) A method ~~of merging interrupt streams of a first type and a second type into a single service routine upon receipt of the interrupt streams by a processor having operationally sequential placement of first type and second type vector addresses, said method~~ comprising the steps of:

merging a first type interrupt and a second type interrupt into a single service routine upon receipt of the first type interrupt and the second type interrupt by a processor having an interrupt vector table for operationally sequential placement and execution of a first type vector address corresponding to the first type interrupt and a second type vector address corresponding to the second type interrupt, the second type interrupt having a higher priority than the first type interrupt, the merging step further comprising the steps of:

(a) providing a common interrupt dispatcher;

(b) inserting ~~an~~ a first instruction at the first type vector address that disables ~~interrupts of~~ the second type interrupt;

(c) inserting a second instruction at the second type vector address~~, an other instruction~~ that branches to the common interrupt dispatcher; and

(d) providing the common interrupt dispatcher with an interrupt routine that processes ~~an~~ the first type interrupt, and then re-enables second type interrupts. ~~of the second type~~.

2. (Currently amended) The method of claim 1, wherein ~~said~~ the inserting step (b) comprises inserting a single instruction at the first vector address that disables the first type and second type interrupts.

3. (Currently amended) The method of claim 2, wherein ~~said~~ the providing step (d) comprises re-enabling the first type interrupt and the second type interrupt.

4. (Currently amended) The method of claim 1, wherein ~~said~~ the providing step (d) further

comprises determining whether ~~a received~~ an interrupt ~~was~~ is of the first type or of the second type.

5. (Currently amended) The method of claim 4, wherein ~~said~~ the providing step (d) comprises checking a mode identifier to identify the type of the interrupt.

6. (Currently amended) The method of claim 4, wherein ~~said~~ the providing step (d) further comprises the steps of

providing a first interrupt handler and a second interrupt handler, and

providing the single service ~~interrupt~~ routine with an instruction that branches the first type interrupts to the first interrupt handler~~, and branches~~ the second type interrupts to the second interrupt handler.

7. (Currently amended) The method of claim 1, wherein the first type interrupts ~~is an~~ are IRQ interrupts, and the second type interrupts ~~is an~~ are FIQ interrupts.

8. (Currently amended) The method of claim 1, wherein the first instruction is a single instruction.

9. (Currently amended) A method of configuring an operating system of an ARM® processor ~~having operationally sequential IRQ and FIQ vector addresses to IRQ and FIQ interrupt streams, said method~~ comprising the steps of:

merging IRQ interrupts and FIQ interrupts into a single service routine upon receipt of the IRQ and FIQ interrupts by a processor having an interrupt vector table for operationally sequential placement and execution of IRQ vector addresses corresponding to the IRQ interrupts and an FIQ vector addresses corresponding to the FIQ interrupts, the merging step further comprising the steps of:

(a) providing a common interrupt dispatcher;

(b) inserting at the IRQ vector address a single instruction that disables an FIQ interrupt

mode;

(c) inserting at the FIQ vector address ~~another~~ a second instruction that branches to the common interrupt dispatcher; and

(d) providing the common interrupt dispatcher with ~~an interrupt~~ the single service routine that processes an interrupt, and the re-enables the FIQ interrupt mode.

10. (Currently amended) The method of claim 9, wherein ~~said~~ the inserting step (b) comprises inserting an instruction at the IRQ vector address that disables an IRQ interrupt mode and the FIQ interrupt mode.

11. (Currently amended) The method of claim 10, wherein ~~said~~ the providing step (d) comprises re-enabling the IRQ and FIQ interrupt modes.

12. (Currently amended) The method of claim 1, wherein ~~said~~ the providing step (d) further comprises determining whether ~~a received~~ an interrupt ~~was~~ is an IRQ interrupt or an FIQ interrupt.

13. (Currently amended) The method of claim 12, wherein ~~said~~ the providing step (d) comprises checking a mode identifier to identify the type of the interrupt.

14. (Currently amended) The method of claim 12, wherein ~~said~~ the providing step (d) further comprises the steps of

providing an IRQ interrupt handler and an FIQ interrupt handler, and

providing the single service ~~interrupt~~ routine with an instruction that branches the IRQ interrupts to the IRQ interrupt handler~~, and branches~~ the FIQ type interrupts to the FIQ interrupt handler.

15. (Currently amended) A method ~~of operating a processor capable of receiving multiple interrupt types, at operationally sequentially placed first type and second type interrupt vector~~

~~addresses, said method~~ comprising the steps of:

operating a processor capable of receiving first type interrupts and second type interrupts, at operationally sequentially placed first type and second type interrupt vector addresses, the first type vector address corresponding to the first type interrupts and the second type vector address corresponding to the second type interrupts, the second type interrupts having a higher priority than the first type interrupts, the operating step further comprising the steps of:

(a) providing a common interrupt dispatcher having an interrupt routine that check a mode identifier to determine whether a received interrupt was of a first and second type, and processes the interrupt;

(b) inserting at the first type vector address ~~an~~ a first instruction that disables subsequent interrupts of the first and second types;

(c) inserting at the second type vector address ~~an other~~ a second instruction that branches to the common interrupt dispatcher;

(d) receiving ~~an~~ a first type interrupt ~~of the first type;~~

~~(e)~~ at the first type vector address;

~~(f)~~ (e) setting the mode identifier to indicate ~~an~~ a first type interrupt ~~of the first type was~~ is received;

~~(g)~~ (f) executing the instruction to disable the first and second type interrupts ~~types~~;

~~(h)~~ (g) executing the other second instruction to branch to the common interrupt dispatcher;

~~(i)~~ (h) processing the first type interrupt ~~of the first type~~ with the common interrupt dispatcher without interruption; and

~~(j)~~ (i) re-enabling the first and second types interrupts ~~types~~.

16. (Currently amended) The method of claim 15, wherein the processor is an ARM® processor and the first type interrupt ~~type~~ is an IRQ interrupt and the second type interrupt ~~type~~ is an FIQ interrupt.

17. (Currently amended) A method ~~of operating a processor capable of receiving multiple~~

~~interrupt types, at operationally sequentially placed first type and second type interrupt vector addresses, said method~~ comprising the steps of:

operating a processor capable of receiving first type interrupts and second type interrupts, at operationally sequentially placed first type and second type interrupt vector addresses, the first type vector address corresponding to the first type interrupts and the second type vector address corresponding to the second type interrupts, the second type interrupts having a higher priority than the first type interrupts, the operating step further comprising the steps of:

(a) providing a common interrupt dispatcher having an interrupt routine that check a mode identifier to determine whether a received interrupt was of a first and second type, and processes the interrupt;

(b) inserting at the second type vector address ~~an other~~ a first instruction that branches to the common interrupt dispatcher;

(c) receiving ~~an~~ a first type interrupt ~~of the first type;~~

~~(d)~~ at the second type vector address;

~~(e)~~ (d) setting the mode identifier to indicate ~~an~~ a second type interrupt ~~of the second type was~~ is received;

~~(f)~~ (e) executing the first instruction to branch to the common interrupt dispatcher; and

~~(i)~~ (h) processing the second type interrupt ~~of the second type~~ with the common interrupt dispatcher without interruption.

18. (Currently amended) The method of claim 17, wherein the processor is an ARM® processor and the first type interrupt ~~type~~ is an IRQ interrupt and the second type interrupt ~~type~~ is an FIQ interrupt.

19. (Currently amended) An operating system ~~for a processor having a facility to handle interrupt streams of a first and a second type upon receipt of the interrupt streams at operationally sequentially placed first type and second type vector addresses said operating system~~ comprising:

a code for merging a first type interrupt and a second type interrupt into a single service routine upon receipt of the first type interrupt and the second type interrupt by a processor having

an interrupt vector table for operationally sequential placement and execution of a first type vector address corresponding to the first type interrupt and a second type vector address corresponding to the second type interrupt, the second type interrupt having a higher priority than the first type interrupt, the code further comprising:

(a) a first ~~an~~ instruction that disables first and second type interrupts ~~types~~, disposed in the first type vector address;

(b) a second ~~an other~~ instruction that branches to a common interrupt dispatcher, disposed at the second type vector address;

(c) the common interrupt dispatcher having an interrupt routine that checks a mode identifier to determine whether a received interrupt was of the first or second type and then re-enables the first and second type interrupts ~~types~~.

20. (Currently amended) The operating system of claim 19, wherein first and second type interrupts ~~of the first and second types~~ are processed by the common interrupt dispatcher without interruption.

21. (Currently amended) A operating system of claim 19, wherein the processor is an ARM® processor, the first type interrupt ~~type~~ is an IRQ interrupt, and the second type interrupt ~~type~~ is and FIQ interrupt.

22. (Currently amended) A method of operating an ARM® processor, comprising the steps of:

(a) providing the system of claim 21;

(b) receiving an IRQ interrupt;

(c) branching to an IRQ vector address;

(d) setting a mode identifier to indicate an IRQ interrupt was received;

(e) executing the first instruction to disable the IRQ and FIQ interrupts;

(f) executing the ~~other~~ second instruction to branch to the common interrupt dispatcher;

(g) processing the IRQ interrupt with the common interrupt dispatcher without interruption; and

(h) re-enabling the IRQ and FIQ interrupts.

23. (Currently amended) A method of operating an ARM® processor, comprising the steps of:

(a) providing the system of claim 21;

(b) receiving an FIQ interrupt;

(c) branching to an FIQ vector address;

(d) setting a ~~the~~ mode identifier to indicate an FIQ interrupt was received;

(e) executing the second instruction ~~at the FIQ vector~~ to branch to the common interrupt dispatcher;

(f) processing the FIQ interrupt with the common interrupt dispatcher without interruption.

24. (Currently amended) An operating system ~~kernel for a processor having a facility to handle interrupt streams of a first and a second type upon receipt of the interrupt streams at operationally sequentially placed first type and second type vector addresses said operating system~~ comprising:

a code for merging a first type interrupt and a second type interrupt into a single service routine upon receipt of the first type interrupt and the second type interrupt by a processor having an interrupt vector table for operationally sequential placement and execution of a first type vector address corresponding to the first type interrupt and a second type vector address corresponding to the second type interrupt, the second type interrupt having a higher priority than the first type interrupt, the code further comprising:

(a) a first ~~an~~ instruction that disables first and second type interrupts ~~types~~, disposed in the first type vector address;

(b) a second ~~an other~~ instruction that branches to a common interrupt dispatcher, disposed at the second type vector address;

(c) the common interrupt dispatcher having an interrupt routine that checks a mode identifier to determine whether a received interrupt was of the first or second type and then re-enables the first and second type interrupts ~~types~~.

25. (Canceled)

26. (Currently amended) An article of manufacture comprising a computer usable medium having a computer readable program code embodied therein, said computer usable medium having:

computer readable program code for providing a common interrupt dispatcher for a processor having:

(a) a first interrupt mode and a second interrupt mode to respectively accept interrupt requests of first and second types, the second interrupt mode having a higher priority that the first interrupt mode, both first and second interrupt modes being disableable to selectively reject interrupt requests of the first and second interrupt types;

(b) a mode status indicator to indicate the current mode of the processor;

(c) a first vector address operatively associated with receipt of interrupt requests of the first type, and a second vector address operatively associated with receipt of interrupt requests of the second type, the first vector address operationally preceding the second vector address in the vector table;

computer readable program code for inserting a first ~~an~~ instruction at the first vector address that disables the first and second interrupt modes;

computer readable program code for inserting a second ~~an other~~ instruction at the second vector address, that branches to the common interrupt dispatcher;

computer readable program code for providing the common interrupt dispatcher with an interrupt routine that checks the mode identifier to determine whether a received interrupt call was a first interrupt or a second interrupt type, the interrupt routine processes the interrupt and re-enables the first and second interrupt modes.

27. (Original) The article of manufacture of claim 26, further comprising:

computer readable program code for providing the interrupt routine with an instruction that branches first interrupts to a first interrupt handler, and branches second interrupts to a second interrupt handler.

28. (Original) The article of manufacture of claim 27, wherein the first interrupt is an IRQ interrupt, and the second interrupt is an FIQ interrupt.

29. (Canceled)

30. (Canceled)

31. (Canceled)

32. (Currently amended) A method of providing a unified interrupt handling system for an embeddable processor having multiple interrupt types, said method comprising the steps of:

(a) providing a processor having a first interrupt mode for accepting interrupt requests of a first type at a first type vector address, and a second interrupt mode for accepting interrupt requests of a second type at a second type vector address, the second interrupt mode having a higher priority than the first interrupt mode, both first and second interrupt modes being selectively disableable to selectively reject interrupt requests of the first and second interrupt requests of the first and second interrupt types;

(b) providing a mode status indicator to indicate the current mode of the processor;

(c) the processor being configured so that the first type vector address operationally precedes the second vector address;

(d) the processor being adapted to execute at least one instruction at the first type and second type vector addresses without interrupt, upon receipt of an interrupt request;

(e) providing a common interrupt dispatcher;

(f) inserting a first ~~an~~ instruction at the first vector address that disables the second interrupt type;

(g) inserting at the second vector address, a second ~~an other~~ instruction that branches to the common interrupt dispatcher; and

(h) providing the common interrupt dispatcher with an interrupt routine that processes the interrupt, and the re-enables the second interrupt types.